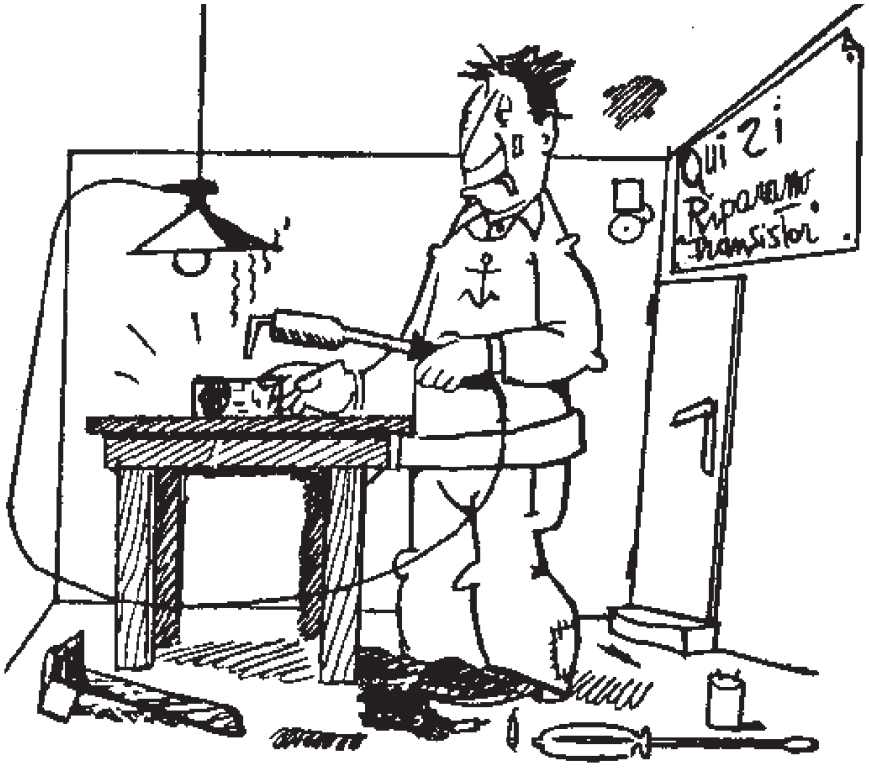


informa@iw1axr.eu

Questo articolo è stato pubblicato su....

radio kit elettronica

Orologio senza display

di Daniele Cappa IW1AXR

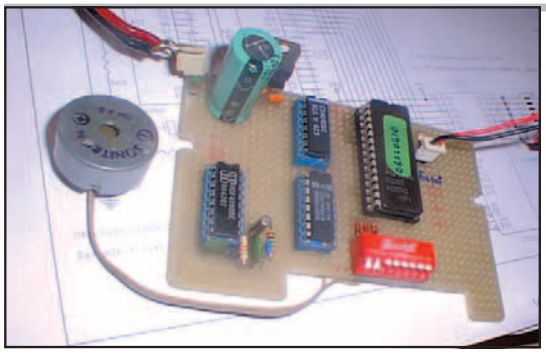


Foto 1 - Il prototipo montato su basetta millefori

Senza alcun tipo di visualizzatore questo orologio comunica come i suoi nonni, suonando un campanello, una campana o qualsiasi altra diavoleria rumorosa. L'elettronica non solo ci permette di stabilire gli orari in cui l'aggeggio deve tacere, ma ci dà la possibilità di riciclarlo come timer da pompa di irrigazione o qualsiasi altra cosa debba essere attivata a cadenza giornaliera, e non solo.

Questa versatilità si paga in sede di progetto, per ogni modifica ai periodi o alle modalità di intervento dobbiamo compilare una immagine e con questa programmare una eprom.

L'idea originale è stata sviluppata per un orologio vero, qualcosa che possa essere preciso quanto è richiesto, che suoni ore e mezze ore con cadenza idonea, che non infastidisca i vicini durante le ore notturne e che possa essere adattato a diversi sistemi di "comunicazione audio". Potremmo persino comandare dispositivi diversi contemporaneamente! Suonare le ore e nello stesso tempo bagnare il giardino!

Il progetto è stato sviluppato prevedendo innumerevoli modifiche, dunque la realizzazione del prototipo è stata eseguita su una basetta millefiori.

Principio di funzionamento

All'insegna del massimo della semplicità si tratta di far suonare le ore e, se lo desideriamo, le mezze ore da una campana. Per programmare le sequenze in cui l'orologio dovrà suonare ci serviamo dei bit di uscita della eprom che viene usata come una grossa tabella utilizzata per memorizzare le azioni che dovrà compiere il trasduttore. Di questa enorme tabella ne viene effettuata una scansione al giorno sfruttando un clock il cui periodo è di 2 secondi.

La partenza è un classicissimo quarzetto da orologi a 32,768 kHz cui fanno seguito una sequenza di divisori tale da portare la frequenza di clock fino a 0.5 Hz con cui vengono pilotati altri divisori le cui uscite rappresentano gli indirizzi della eprom da 64Kb.

Per meglio comprendere il funzionamento dell'orologio è necessario avere un pochino di famigliarità con i numeri esadecimali (a base 16).

Un giorno è composto da 86400 secondi (60 secondi per 60 minuti per 24 ore), con un clock il cui periodo è pari a 2 secondi abbiamo la corrispondenza di 43200 impulsi di clock in 24 ore, ecco che una semplice eprom da 64Kb copre tutta la giornata. Con questo clock un'ora occupa 1800 periodi (la metà di 3600, ovviamente), $0708 in esadecimale. La nostra eprom andrà programmata in modo che ogni evento che definiamo a cadenza oraria sia programmato dall'indirizzo $0000, ogni $0708 e suoi multipli; gli eventi che dovranno essere eseguiti allo scadere della mezzora saranno spostati in avanti di 900 byte, $0384 in esadecimale.

Con questo sistema abbiamo la possibilità di realizzare un orologio estremamente versatile: abbiamo sette canali rappresentati dai primi sette bit di uscita della eprom (l'ottavo bit è usato internamente dall'orologio per resettare il contatore allo scadere delle 24 ore) con cui comandare praticamente qualsiasi cosa. Le uniche limitazioni sono la ripetitività giornaliera e la lunghezza di ogni evento che non potrà essere inferiore a un impulso di clock.

Veniamo al nostro orologio, i bit d'uscita pilotano, tramite normali diodi al silicio, un transistor che a sua volta comanda l'elettrocalamita di un robusto ex_relè il cui compito è di dare opportuni colpi ad una campana. Ecco dunque ottenuto il nostro scopo che era di "suonare le ore".

Il prototipo monta un semplice cicalino a cialda piezo alimenta-

to direttamente dai bit di uscita della eprom (Foto 1), per effettuare i test senza disturbare nessuno.

Fino a qui è stato fatto un bel discorso in termini di tempo, di periodi che intercorrono tra un evento e il successivo, ma come faccio a regolare questo coso che ci si ostina a chiamare orologio?

Il conteggio giornaliero termina allo scadere del 43200esimo impulso, dunque l'ultimo byte utile della eprom sarà quello all'indirizzo $A8C0, a questo punto la eprom dovrà attivare l'ultimo bit di uscita (O7) che passando a livello 1 resetta i due CD4040 che ripartiranno a contare gli impulsi da zero. Abbiamo simulato il passaggio della mezzanotte, o meglio della fine del ciclo di 24 ore e il ripristino delle condizioni iniziali.

La messa in passo dell'orologio avviene… resettandolo a mano, tramite P1, allo scadere di quelle che abbiamo considerato come l'inizio del ciclo di 24 ore. Per comodità ho preparato la eprom con le ore 13.00 come inizio del giorno del nostro stranissimo orologio; del resto data l'assenza di display, lancette o altro non vi era molta scelta!

Prepariamo il file della eprom

Questa è l'operazione più impegnativa, non perché sia difficile, ma perché è necessario comprendere bene cosa dobbiamo inserire nella eprom e perché.

Prepariamo il file "a mano" con il programma in dotazione al nostro programmatore di eprom, o con Icprog 1.04 semplicemente settando il chip come 24C512, poi dal menù "modifica" è necessario riempire il buffer con "0".

Attenzione! Il programma Icprog è molto utile per compilare il file immagine della eprom, ma tra le sue funzioni NON è prevista la programmazione della eprom usata! Il chip 24C512 con la 27C512 che stiamo utilizzando ha in comune solamente la lunghezza del file immagine che dovrà essere usato per la sua programmazione utilizzando un programmatore idoneo e il software che lo accompagna.

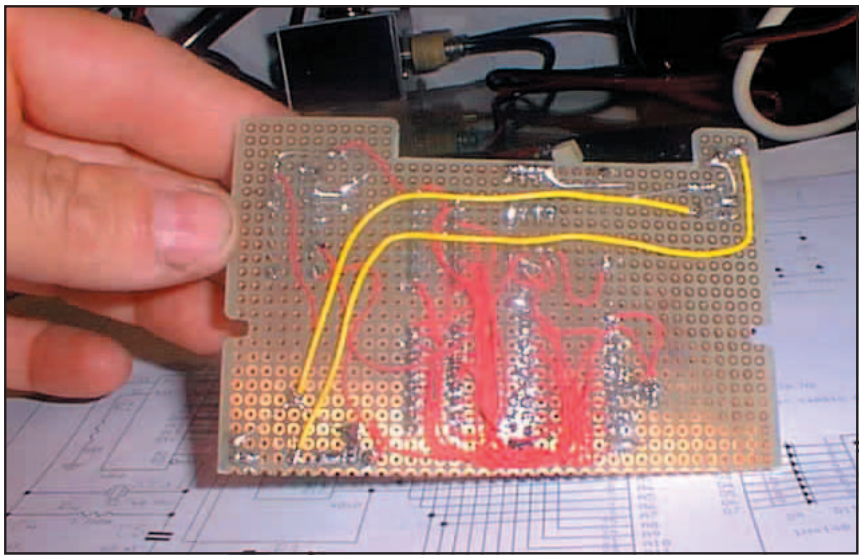


Foto 2 - Lato saldature del prototipo

Quindi procedere a riportare nella eprom la sequenza che desideriamo. Le uscite della eprom saranno attive quando il bit corrispondente è attivo: se alla locazione $0708 poniamo 1 allora solo il primo bit sarà a livello 1 quando il contatore ci passerà sopra. Se vi mettiamo un 89 (137 in decimale e 10001001 in binario) allora avremo a uno i bit di uscita di peso 1, 8 e 128 (O0, O3 e O7 1+8+128=137).

Se vogliamo avere una uscita alta per 45 minuti (45 per 60 secondi sono 2700 secondi) dovremmo porre a uno il bit interessato per un periodo pari a 1350 impulsi di clock; vale a dire da una locazione $xxxx alla locazione $xxxx + $0546 (1350 in esadecimale vale $0546)

Ecco alcune righe del file della eprom dell'orologio, il primo numero, quello prima dei ":", è l'indirizzo del primo byte degli otto visualizzati:

0000: **0005** 0000 0000 0000 0000 0000 0000 0000

Il primo rintocco, le ore 13.00 alla locazione 0000 sul bit 0 e bit 2 (1+4)

….

0380: 0000 0000 0000 0000 **000A** 0000 0000 0000

La prima ½ ora sul bit 1 e bit 3 (2+8) sul byte $0384

….

0708: **0005** 0000 **0005** 0000 0000 0000 0000 0000

I due rintocchi delle 14.00 sempre su bit 0 e bit 2

Il file è strutturato per avere sul bit 0 i rintocchi e sul bit 1 le mezze ore, facendo suonare l'orologio solo dalle 8.30 del mattino alle 21.

Sul bit 2 e bit 3 rispettivamente i rintocchi e le mezze ore senza la pausa notturna.

Se per comandare qualsiasi cosa desideriamo attivare il bit 4 dalle 3 alle 5 del mattino dobbiamo riempire tutti i byte compresi dall'indirizzo $6270 a $7080 aggiungendo $10 (16 in decimale) al valore attuale.

Ecco come siamo arrivati a questo:

Le 3 del mattino sono 14 ore dopo l'inizio del nostro giorno (che inizia alle ore 13, è strano, ma a noi va bene così), 1800 impulsi all'ora per 14 ore sono 25200 impulsi, in esadecimale $6270. Da questa posizione dobbiamo attivare il bit che ci interessa scrivendo il suo valore in tutti i byte seguenti, sommandolo al valore che eventualmente ci fosse già. Lo stop avverrà due ore dopo, 3600 impulsi dopo, quando abbiamo raggiunto il numero

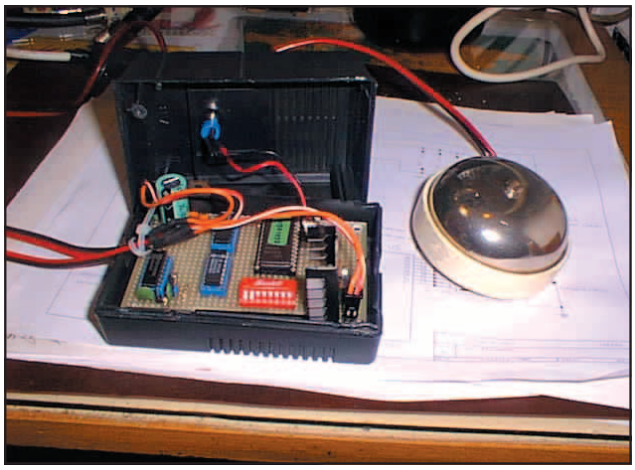


Foto 3 -Ultimi test con un campanello

Foto 4 - Finito!

28800 (25200 + 3600), in esadecimale $7080.

Questi calcoli si potranno eseguire con una normale calcolatrice scientifica in grado di eseguire conversioni da decimale a esadecimale, oppure con la calcolatrice di windows.

Il riempimento del buffer con un valore fisso potrà essere eseguito direttamente con Icprog, semplificando l'operazione.

Al termine attiviamo nell'ultima locazione in uso l'ultimo bit, quello che dovrà resettare il conteggio, il valore decimale è 128, $80 in esadecimale:

A8C0: **0080** 0000 0000 0000 0000 0000 0000 0000

Possiamo intervenire sulla precisione dell'orologio spostando di uno o più byte avanti o indietro il punto in cui il bit 7 di uscita della eprom resetta i contatori. Spostando il $80 verso sinistra compensiamo se l'orologio ritarda mentre spostando verso destra compensiamo se l'orologio anticipa. Ogni byte anticipa o ritarda il conteggio di 2 secondi.

L'errore accumulato nel giorno è così compensato allungando o accorciando il conteggio dell'ultima ora.

Il file andrà salvato con estensione.BIN affinché sia compatibile con la maggior parte dei programmatori di eprom.

Ovviamente il file che ho utilizzato è disponibile per i lettori.

Montaggio e scelta componenti

Il tutto è stato montato su basetta millefori (Foto 2), senza il comando del relè della campana l'orologio assorbe 8 mA, cosa che permette di alimentarlo prevedendo anche una piccola batteria di backup, magari un elemento NiCd da 9 V con cui l'orologio potrebbe sopravvivere una dozzina di ore senza alimentazione. Un modello più robusto, al piombo per antifurto, potrebbe alimentarlo per un mese.

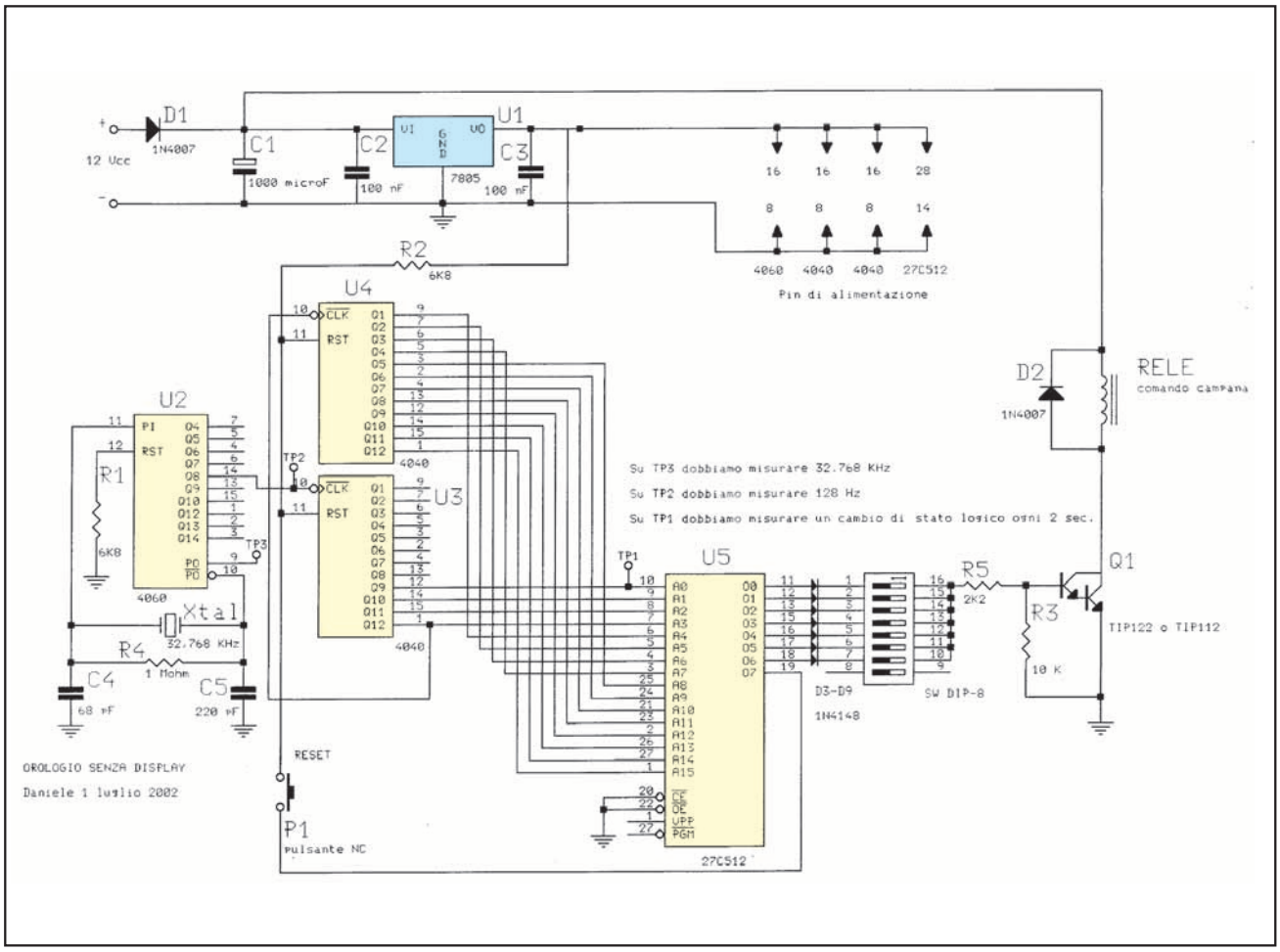
I componenti usati sono tutti di tipo Cmos della serie CD40xx, lo stabilizzatore dovrebbe essere un 78L05, il prototipo monta un 7805 da 1 A che aumenta un poco l'assorbimento di corrente. Il quarzo è nel contenitore classico da orologio.

La eprom è una 64 Kbyte, preferibilmente in tecnologia Cmos, 27C512 più facilmente programmabile di un modello da 128K con cui avremmo potuto ottenere periodi di un secondo. Lo scopo dell'orologio non è danneggiato dalla lunghezza del periodo minimo pari a 2 secondi, se non con una maggiore spaziatura nei rintocchi orari che si fa maggiormente apprezzare se usato in unione a una campana classica.

Il dip SW a 8 vie, di cui solo 7 sono usati, decide quale dei bit di uscita è utilizzato, è possibile usare più bit contemporaneamente, nella eprom utilizzata sul prototipo sono usati insieme il bit 0 e il bit 1 rispettivamente per le ore e per le ½ ore utilizzando i bit che prevedono la pausa notturna.

Il test dell'orologio richiede tempi di attesa piuttosto lunghi, si può ovviare in parte spostando l'ingresso del clock di U3, il primo 4040, dal pin 14 al pin 7 del 4060. In queste condizioni riduciamo tutti i tempi di 16 volte: le 24 ore si svolgono ora in 90 minuti, un'ora dura 3' e 45", la ½ ora dura 1' e 52.5". Possiamo controllare che i tempi che abbiamo stabilito siano rispettati senza perdere settimane in test. Successivamente è necessario riportare tutto alle condizioni iniziali per le prove di precisione sulle 24 ore "vere".

La campana andrebbe realizzata con un tubo di acciaio, o di bronzo, lungo 77 cm appeso per una estremità e percosso dal basso da un martellino azionato dal ritorno dell'ancora di un robusto ex relè. Il coso così realizzato dovrebbe produrre un "LA" che mi dicono adatto all'uso su un orologio. Il consulente del caso è rappresentato dal Mauro Brignolo IK1OVY, di professione suonatore di Trombone, che qui pubblicamente ringrazio. Più facilmente si può recuperare un campanello a 12V (foto 3) oppure un campanello a Din-Don, cui è necessario eliminare la piastrina del "Din" affinché non siano



suonati due rintocchi al posto di uno!

Il contenitore del prototipo è un recupero ed è stato installato a metà luglio del 2002, collocando l'unità orologio in stazione e il campanello sul tetto utilizzando la configurazione visibile nella foto 4, i due sono uniti da una coppia di cavi preesistenti.

I costi sono più che contenuti, escludendo la campana, per cui ognuno utilizzerà quel che ha a disposizione, il tutto supera di poco i 10 euro. Il quarzo può essere recuperato da un orologio digitale, anche da polso, così come dalla scheda madre di un PC. I tre Cmos sono roba da 50 centesimi e il darlington è un ex_sirena antifurto.

Possibile modifiche…

La versatilità va ancora oltre, sfruttando tutti i divisori contenuti nei CD4060 e prelevando il clock del primo 4040 dal pin 3 del 4060 possiamo allungare il periodo da uno a 64 giorni, con un periodo minimo di intervento che passa da 2 a 128 secondi. Modificando l'indirizzo in cui è programmato il valore di reset possiamo adattare il tutto ad un nuovo periodo di conteggio.

Il relè e il suo comando può essere realizzato in più esemplari se riteniamo utile il comando separato per ogni bit della eprom.

Abbiamo realizzato un orologio che, all'occorrenza, diventa un timer ciclico con un periodo di oltre due mesi… sempre considerando che il conteggio è di tipo binario, dunque i tempi che otteniamo sono sempre "il doppio" oppure "la metà" di quelli ottenuti utilizzando il bit di peso maggiore, o minore.

Rimanendo su tempi lunghi possiamo ottenere un ciclo ogni 64 giorni, oppure uno di 32 …

Lista componenti

R1 =6K8
R2 = 6K8
R3 = 10K
R4 = 1MΩ
R5 = 2K2
C1 = 1000 μF
C2 = 100 nF
C3 = 100 nF
C4 = 68 pF
C5 = 68 pF
D1 = 1N4007
D2 = 1N4007
D3 – D9 = 1N4148
Q1 = TIP112 o TIP122
U1 = 78L05
U2 = CD4060
U3 = CD4040
U4 = CD4040
U5 = eprom 27C512 programmata
P1 = pulsante di reset, normalmente CHIUSO
SW = dip SW a 8 vie
3 = zoccoli a 16 pin
1 = zoccolo a 28 pin
Xtal = quarzo miniatura da orologio a 32,678 kHz

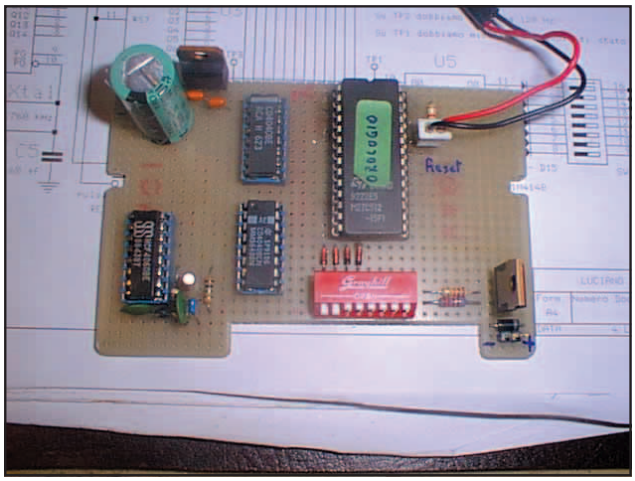


Foto 5 - Il cicalino è stato sostituito dal darlington

Foto 6 - Montaggio provvisorio

ma non uno di 40!... a meno di non modificare la frequenza del quarzo.

I transistor usati per il comando del relè sono due NPN darlington rispettivamente da 2 e da 5 A (Foto 5), cosa che ci permette di comandare carichi di tutto rispetto, sono comunque sostituibili con modelli più modesti se dobbiamo pilotare un piccolo relè tipo feme da 5 A sui contatti.

Se non vogliamo, o non possiamo, intervenire su quello che potremmo considerare il software dell'orologio per ritoccare il periodo di conteggio possiamo sostituire il condensatore C5 con un compensatore da 150 pF per intervenire sulla frequenza del quarzo, che possiamo misurare su TP3, ovvero sul pin 9 del CD4060.